Bioinformatics analysis task

Applicant: Carlos Neves

Company: Sophia Genetics

Task description:

In this report I analyse the NGS reads for human DNA regions using a pair-ended amplicon sequencing design on a Miseq sequencing platform. The data was provided by Sophia Genetics as the compressed file sequenceFile.zip, which contains the raw fastq reads file and the alignment (hg19 human genome) file in BAM format.

The primers used for sequencing enrichment were

```
sequences <- list(
    "primer_forward_1" = "TTGCCAGTTAACGTCTTCCTTCTCTCTCTG",
    "primer_reverse_1" = "GAGAAAAGGTGGGCCTGAGGTTCAGAGCCA",
    "primer_forward_2" = "CCCTTGTCTCTGTGTTCTTGTCCCCCCCA",
    "primer_reverse_2" = "CCCCACCAGACCATGAGAGGCCCTGCGGCC",
    "primer_forward_3" = "TGATCTGTCCCTCACAGCAGGGTCTTCTCT",
    "primer_reverse_3" = "TGACCTAAAGCCACCTCCTTA",
    "primer_forward_4" = "CACACTGACGTGCCTCTCCCTCCCTCCA",
    "primer_reverse_4" = "CCGTATCTCCCTTCCCTGATTA",
    "adapter_1" = "AAGACTCGGCAGCATCTCCA",
    "adapter_2" = "GCGATCGTCACTGTTCTCCA")
```

Questions:

Please note that I have focused on solving the following questions using the R language as much as possible to unformize and facilitate the generation of a PDF report using Rmarkdown and knitr package. Alternative methods are available and it is not uncommon to use various languages/software to solve these problems.

1. **What is the constitution of each read, for example (adaptor + primer + amplified region)?**

Answer: Each read will be constituted by *forward primer + amplicon + reverse primer*. However, because the read length is not long enough to span the entire amplicon, these reads will not correspond to the full amplicon sequence. The adapters are not present in the read as these are already hybridized (double-stranded) and thus are not "synthetized" during sequencing.

Confirmation: Using the ShortRead library I confirmed the presence/absence of the primers and adapters from the raw reads from the FASTQ files *aln1.fastq.gz* and *aln2.fastq.gz*.

```
# Load required package
library("ShortRead")
```

```
# Read the the FASTQ files into memory
aln_1 <- readFastq("sequenceFile/aln1.fastq.gz")
aln_2 <- readFastq("sequenceFile/aln2.fastq.gz")

# Check forward reads
aln_1
## class: ShortReadQ
## length: 575002 reads; width: 151 cycles
```

```
sread(aln_1)
##   A DNAStringSet instance of length 575002
##          width seq
##      [1]   151 TTGCCAGTTAACGTCTTCCTTCTCTCTCT...GAAATCCTCGATGTGAGTTTCTGCTTTG
##      [2]   151 CACACTGACGTGCCTCTCCCTCCCTCCAG...TGCCCTTCGGCTGCCTCCTGGACTATGT
##      [3]   151 TGGCCAGTTAACGTCTTCCTTCTCTCTCT...AGTTTCTGCTTTGCTGTGTGGGGGTCCA
##      [4]   151 TTGCCAGTTAACGTCTTCCTTCTCTCTCT...GAAATCATCGATGAGAGTTTCTGCTTTG
##      [5]   151 TTGCCAGTTAACGTCTTCCTTCTCTCTCT...GAAATCCTCGATGTGAGTTTCTGCTTTG
##      ...   ... ...
## [574998]   151 CACACTGACGTGCCTCTCCCTCCCTCCAG...TGACCTTCGGCTGCCTCCTGGACTATGT
## [574999]   151 TTGCCAGTTAACGTCTTCCTTCTCTCTCT...GAAATCCTCGATGTGAGTTTCTGCTTTG
## [575000]   151 TTGCCAGTTAACGTTTTCCTTCTCTCTCT...AGTTTCTGCTTTGCTGTGTGGGGGTCCA
## [575001]   151 TGAATCTGTCCCTCACAGCACCCTCACCC...CTGGGCTCCAGGGTACAGTGTCCTCCAG
## [575002]   151 CACACTGACGTGCCTCTCCCTCCCTCCAG...TGCCCTTCGGCTGCCTCCCGGACTATGT

# Check reverse reads
aln_2
## class: ShortReadQ
## length: 575002 reads; width: 151 cycles
sread(aln_2)
##   A DNAStringSet instance of length 575002
##          width seq
##      [1]   151 NAGAAAAGGTGGGCCTGAGGTTCAGAGCC...GAATTTTAACTTTCTCACCTTCTGGGAT
##      [2]   151 NCGTGTCTCCCTTCCCGGATTACCTTTGC...GGGATGAGTGGCACGGTGGAGGTGAGGG
##      [3]   151 NAGAAAAGGGGGGCCTGAGGTTCAGAGCC...CACCTTCTGGGATCCAGAGTCCCTATGG
##      [4]   151 NAGAAAAGGTGGGCCTGAGGTTCAGAGCC...GAATTTTAACTTTCTCACCTTCTGGGCT
##      [5]   151 GAGAAAAGGTGGGCCTGAGGTTCAGAGCC...GAATTTTAACTTTCTCACCTTCTGGGAT
##      ...   ... ...
## [574998]   151 CGGTATCTCCCGTCCCTGATTACCTTTGC...GTGATGAGTTGCACGGTGGAGGTGAGGG
## [574999]   151 GAGAAAAGGTGGGCCTGAGGTTCAGAGCC...GAATTTTAACTTTCTCACCTTCTGGGAT
## [575000]   151 GAGAAAAGGTGGGCCTGAGGTTCAGAGCC...CACCTTCTGGGATCCAGAGTCCCTATGG
## [575001]   151 GAAAAAGGTGGGCCTTAACGTCAGAAACA...CACTGTACCCTGGAGCCCCGGCGGGCCT
## [575002]   151 CCGTATCTCCCTTCCCTGATCACCTTTGC...GTGATGAGCTGCACGGTGGGGGTGAGGG


# Look for adapters and primers in reads
mismatch <- round(mean(sapply(sequences,nchar))*0.1)

for (idx in 1:length(sequences)){
    count_1 <- sum(vcountPattern(sequences[[idx]], sread(aln_1),  max.mismatch=mismatch)==1)
    init_1 <- length(grep(paste0("^", sequences[[idx]]), sread(aln_1)))

    count_2 <- sum(vcountPattern(sequences[[idx]], sread(aln_2),  max.mismatch=mismatch)==1)
    init_2 <- length(grep(paste0("^", sequences[[idx]]), sread(aln_2)))


    cat(paste("Sequence:", names(sequences)[idx], "\n",
    "In aln1:", count_1, "of which", init_1, "perfect matches to begining\n",
    "In aln2:", count_2, "of which", init_2, "perfect matches to begining\n\n"))
}
## Sequence: primer_forward_1
##  In aln1: 218831 of which 203060 perfect matches to begining
##  In aln2: 0 of which 0 perfect matches to begining
```

```
##
## Sequence: primer_reverse_1
##  In aln1: 0 of which 0 perfect matches to begining
##  In aln2: 215535 of which 182627 perfect matches to begining
##
## Sequence: primer_forward_2
##  In aln1: 9147 of which 6430 perfect matches to begining
##  In aln2: 0 of which 0 perfect matches to begining
##
## Sequence: primer_reverse_2
##  In aln1: 0 of which 0 perfect matches to begining
##  In aln2: 9035 of which 8052 perfect matches to begining
##
## Sequence: primer_forward_3
##  In aln1: 106303 of which 97002 perfect matches to begining
##  In aln2: 0 of which 0 perfect matches to begining
##
## Sequence: primer_reverse_3
##  In aln1: 8 of which 0 perfect matches to begining
##  In aln2: 107968 of which 90419 perfect matches to begining
##
## Sequence: primer_forward_4
##  In aln1: 155130 of which 144146 perfect matches to begining
##  In aln2: 0 of which 0 perfect matches to begining
##
## Sequence: primer_reverse_4
##  In aln1: 0 of which 0 perfect matches to begining
##  In aln2: 171925 of which 128507 perfect matches to begining
##
## Sequence: adapter_1
##  In aln1: 5399 of which 0 perfect matches to begining
##  In aln2: 1 of which 0 perfect matches to begining
##
## Sequence: adapter_2
##  In aln1: 12 of which 0 perfect matches to begining
##  In aln2: 8 of which 0 perfect matches to begining
```

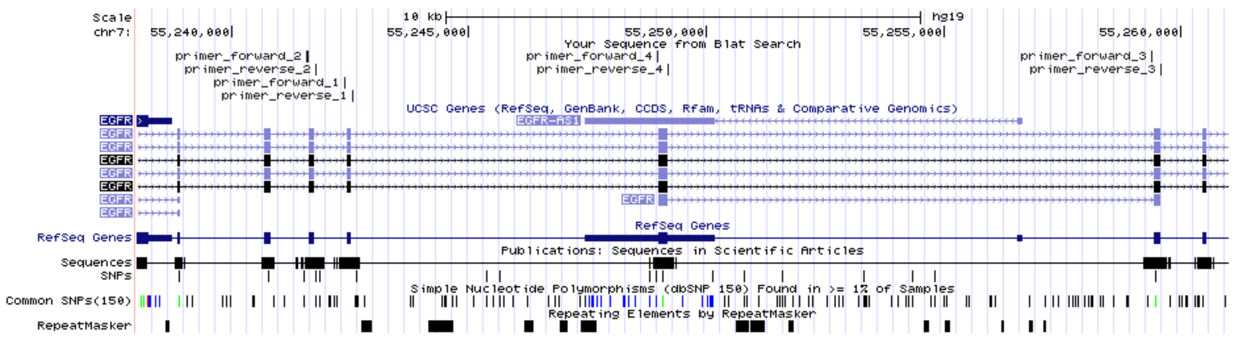
Few adapters are found in the FASTQ files, while the primers commonly match perfectly to the read start. For aln1.fastq there are 489 411 reads with one of the primers (450 638 at the read start) out of 575 002 reads, suggesting 78-85% of the reads may correspond to the expected sequences. For aln2.fastq there are 504 463 reads with one of the primers (409 605 at the read start) out of 575 002 reads, suggesting 71-88% of the reads may correspond to the expected sequences.

Additional exploration: Since there were only 4 primer pairs, I converted the primer sequences to FASTA format and checked the expected amplicons on UCSC Genome Browser

```
null_return <- sapply(grep("primer", names(sequences)), function(idx) {
    cat(paste("> ", names(sequences)[idx], "\n",
    sequences[[idx]], "\n\n"))
})
## >  primer_forward_1
##  TTGCCAGTTAACGTCTTCCTTCTCTCTCTG
##
## >  primer_reverse_1
##  GAGAAAAGGTGGGCCTGAGGTTCAGAGCCA
```

```
##
## >  primer_forward_2
##  CCCTTGTCTCTGTGTTCTTGTCCCCCCCA
##
## >  primer_reverse_2
##  CCCCACCAGACCATGAGAGGCCCTGCGGCC
##
## >  primer_forward_3
##  TGATCTGTCCCTCACAGCAGGGTCTTCTCT
##
## >  primer_reverse_3
##  TGACCTAAAGCCACCTCCTTA
##
## >  primer_forward_4
##  CACACTGACGTGCCTCTCCCTCCCTCCA
##
## >  primer_reverse_4
##  CCGTATCTCCCTTCCCTGATTA
```

The resulting lines of *FASTA* were copied into the UCSC Genome Browser's Blat tool. The resulting image shows the result, illustrating the position of all the primers to the 3' exons for the EGFR gene.



,

All of the amplicons of interest are located at the 3' region of EFGR. To better describe each amplicon, each primer pair was tested using the UCSC Genome Browser's *In-silico* PCR tool.

The expected amplicons are:

- Primer pair 1: chr7:55242379+55242574 196bp

```
amplicons <- toupper(paste0(
  "TTGCCAGTTAACGTCTTCCTTCTCTCTCTGtcatagggactctggatcccaga",
  "aggtgagaaagttaaaattcccgtcgctatcaaggaattaagagaagcaacatc",
  "tccgaaagccaacaaggaaatcctcgatgtgagtttctgctttgctgtgtgggg",
  "gtccaTGGCTCTGAACCTCAGGCCCACCTTTTCTC"))
```

- Primer pair 2: chr7:55241584+55241796 213bp

```
amplicons <- c(amplicons, toupper(paste0(
  "CCCTTGTCTCTGTGTTCTTGTCCCCCCCAgcttgtggagcctcttacacccagt",
  "ggagaagctcccaaccaagctctcttgaggatcttgaaggaaactgaattcaaaa",
  "agatcaaagtgctgggctccggtgcgttcggcacggtgtataaggtaaggtccct",
  "ggcacaggcctctgggctgGGCCGCAGGGCCTCTCATGGTCTGGTGGGG")))
```

- Primer pair 3:chr7:55259375+55259589 215bp

```
amplicons <- c(amplicons, toupper(paste0(
  "TGATCTGTCCCTCACAGCAGGGTCTTCTCTgtttcagggcatgaactacttgga",
  "ggaccgtcgcttggtgcaccgcgacctggcagccaggaacgtactggtgaaaaca",
  "ccgcagcatgtcaagatcacagattttgggctggccaaactgctgggtgcggaag",
  "agaaagaataccatgcagaaggaggcaaagTAAGGAGGTGGCTTTAGGTCA")))
```

- Primer pair 4: chr7:55248957+55249194 238bp

```
amplicons <- c(amplicons, toupper(paste0(
  "CACACTGACGTGCCTCTCCCTCCCTCCAggaagcctacgtgatggccagcgtgga",
  "caacccccacgtgtgccgcctgctgggcatctgcctcacctccaccgtgcagctca",
  "tcacgcagctcatgcccttcggctgcctcctggactatgtccgggaacacaaagac",
  "aatattggctcccagtacctgctcaactggtgtgtgcagatcgcaaaggTAATCAG",
  "GGAAGGGAGATACGG")))
```

Using the predicted sequence for each amplicon, I estimated the edit distance between these and the experimental reads (taking into account the read size of 151 bp). The total number of reads with a edit distance less than 5 were estimated to access how many reads correspond to the expected sequences.

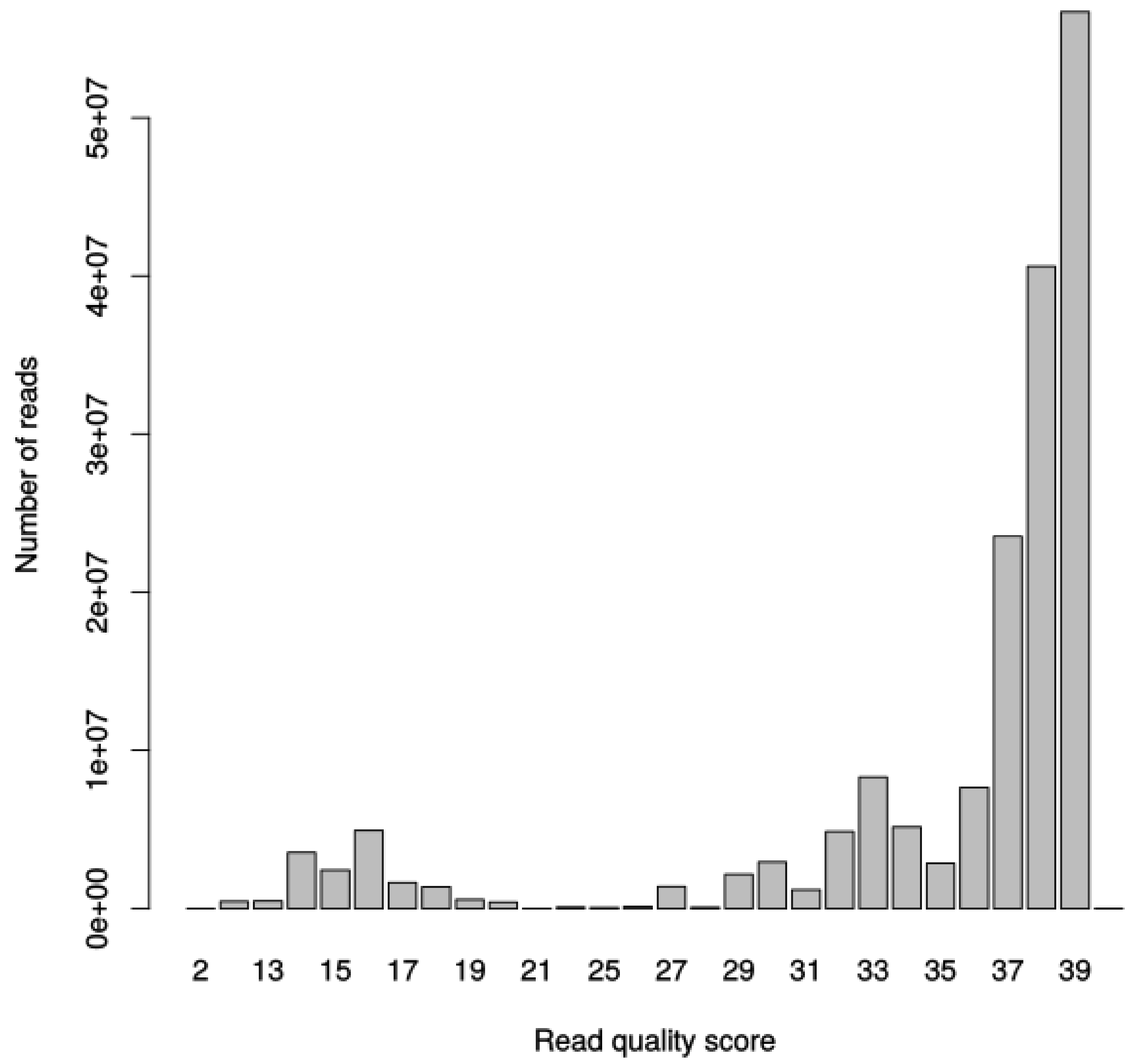
```
# Convert the amplicon sequence vectors to a DNAStringSet object
amplicons <- DNAStringSet(amplicons)

# Estimate the forward distances to each amplicon
for_dist <- srdistance(sread(aln_1), substr(amplicons, 1, 151))

# Estimate the reverse distances to the reverse comliment of each amplicon
rev_dist <- srdistance(sread(aln_2),
                   substr(reverseComplement(amplicons), 1,151))

# Generate all possible distances (names) smaller than 5
dists <- as.character(seq(0,5,0.25))

null_return <- sapply(1:length(for_dist), function(idx) {
    cat(paste("Primer pair", idx,
    "\nNumber of sequences with distance < 5 on aln1:\n",
    sum(table(for_dist[[idx]])[dists], na.rm=TRUE),
    "\nNumber of sequences with distance < 5 on aln2:\n",
    sum(table(rev_dist[[idx]])[dists], na.rm=TRUE), "\n\n"))
})
## Primer pair 1
## Number of sequences with distance < 5 on aln1:
##  109663
## Number of sequences with distance < 5 on aln2:
##  105284
##
## Primer pair 2
## Number of sequences with distance < 5 on aln1:
##  5282
## Number of sequences with distance < 5 on aln2:
##  8870
##
## Primer pair 3
## Number of sequences with distance < 5 on aln1:
##  106465
## Number of sequences with distance < 5 on aln2:
##  100410
##
## Primer pair 4
## Number of sequences with distance < 5 on aln1:
##  152931
## Number of sequences with distance < 5 on aln2:
##  137936
```

Comparing the predicted amplicons to the experimental reads I found that about 374 341 reads (65%) match the expected forward amplicon and 352 500 reads (61%) match the expected reverse amplicon, with an edit distance of 5 or less.

Finally, I did a simple QC for base quality loss at the end of the reads.

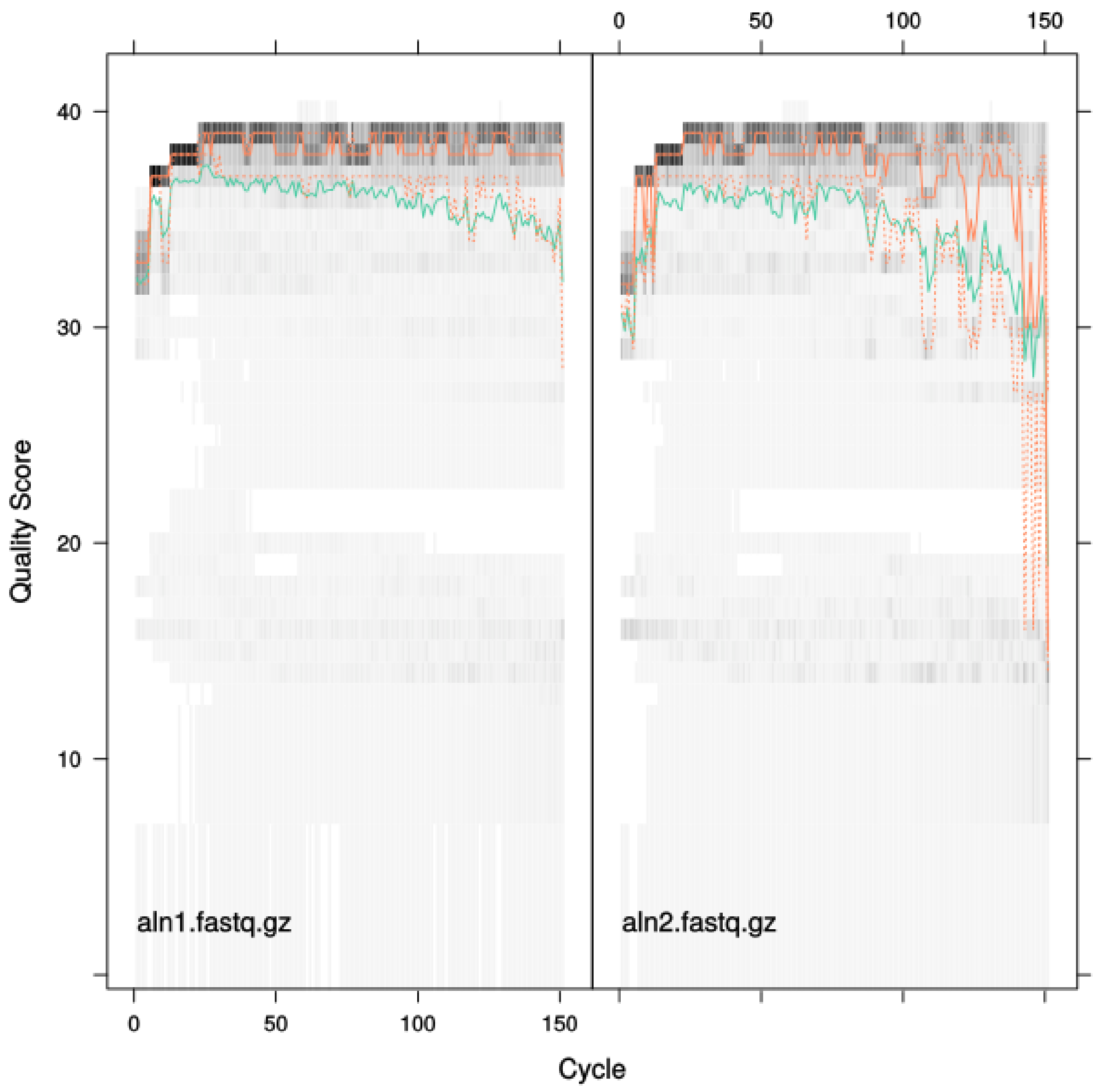
```
# Load required package
library(dplyr)
```

```
# Run QC function for FASTQ files
QC <- qa(c("sequenceFile/aln1.fastq.gz","sequenceFile/aln2.fastq.gz"), type="fastq")

# Check base quality per "amplicication cycle" (position)
base_quality <- QC[["perCycle"]]$quality %>%
  dplyr::select(Score, Count) %>%
  group_by(Score) %>%
  summarise(sum = sum(Count))

as.data.frame(base_quality)
##    Score      sum
## 1      2     6016
## 2     12   457975
## 3     13   486337
## 4     14  3558114
## 5     15  2420571
## 6     16  4935758
## 7     17  1637315
## 8     18  1376991
## 9     19   579042
## 10    20   409495
## 11    21    12121
## 12    24   100427
## 13    25    66676
## 14    26   131608
## 15    27  1386645
## 16    28    80602
## 17    29  2155801
## 18    30  2931502
## 19    31  1184245
## 20    32  4864969
## 21    33  8309364
## 22    34  5157273
## 23    35  2845985
## 24    36  7660305
## 25    37 23543991
## 26    38 40627315
## 27    39 56702352
## 28    40    21809

# Histogram of overall quality
barplot(base_quality$sum, names.arg = base_quality$Score,
        xlab = "Read quality score", ylab = "Number of reads")
```



,

```
# Quality at each "cycle"
ShortRead:::.plotCycleQuality(QC[["perCycle"]]$quality)
```



,

2. **Which gene region did those reads mapped to ?**

Answer: Most of the reads aligned to chromosome 7 (86%), as it would be expected, since the primers are targeting EGFR exons, a gene found in this chromosome. Of all the reads mapping to chromosome 7, 50% show some mapping overlap with the four amplicons considered.

Confirmation: Using the Rsamtools package to read the BAM file, I initially looked into the BAM file specifications.

```
# Load required package
library(Rsamtools)
```

```
scanBAMHeader("sequenceFile/aln.bam")[[1]]$text$"@HD"
## Error in eval(expr, envir, enclos): could not find function "scanBAMHeader"
scanBamHeader("sequenceFile/aln.bam")[[1]]$text$"@RG"
## [1] "ID:Bordet_EGFR_S1" "SM:S1"             "PL:Illumina"
```

```
scanBamHeader("sequenceFile/aln.bam")[[1]]$text$"@PG"
## [1] "ID:bowtie2" "PN:bowtie2" "VN:2.0.2"

# Sort Bam file
sortBam("sequenceFile/aln.bam", "sequenceFile/aln_sorted")
## Warning in .local(file, destination, ...): [bam_sort_core] merging from 2
## files...
## [1] "sequenceFile/aln_sorted.bam"
```

The provided BAM file was obtained using the bowtie2 2.0.2 aligner software and it is unsorted. To prevent any downstream issues with the genomic alignments, I sorted the BAM file. Of note, the Read Group ID for the Illumina platform is Bordet_EGFR_S1, again suggesting EGFR as the gene of interest in this study. I then looked into the BAM alignment statistics to find to which region (chromosome) the reads where aligned to.

```
# BAM alignmnet statistics
bam_stats <- idxstatsBam("sequenceFile/aln_sorted.bam", index="sequenceFile/aln.bam.bai")
bam_stats[1:30,]
##      seqnames seqlength mapped unmapped
## 1           1 249250621   1543      147
## 2           2 243199373   1092      156
## 3           3 198022430   1050      121
## 4           4 191154276  29665      228
## 5           5 180915260    845       98
## 6           6 171115067   1031      127
## 7           7 159138663 993192     1472
## 8           8 146364022   1125       95
## 9           9 141213431    455       73
## 10         10 135534747   1632       84
## 11         11 135006516    969       71
## 12         12 133851895    390       85
## 13         13 115169878     72       48
## 14         14 107349540    169       42
## 15         15 102531392    220       70
## 16         16  90354753   2959       98
## 17         17  81195210    495       68
## 18         18  78077248    680       37
## 19         19  59128983   8894       63
## 20         20  63025520    632      213
## 21         21  48129895    503       22
## 22         22  51304566   3471       38
## 23          X 155270560   1866      117
## 24          Y  59373566     16        8
## 25         MT     16569      0        0
## 26 GL000191.1    106433      0        0
## 27 GL000192.1    547496      1        1
## 28 GL000193.1    189789      0        0
## 29 GL000194.1    191469      0        0
## 30 GL000195.1    182896      2        1

#Number of mapped reads
sum(bam_stats$mapped)
## [1] 1052983
```

993 192 out of the 1 150 004 reads (575 002 from each FASTA) were aligned to chromosome 7, representing

86% of all sequenced reads and 94% of all aligned reads. Since amplicon sequence was targeted to the EGFR locus, I additionally looked into the chromosome 7 alignment.

```
# Read BAM file
bam <-scanBam("sequenceFile/aln_sorted.bam",
   index =  "sequenceFile/aln.bam.bai")

# Convert BAM file to dataframe
bam_ranges <- as.data.frame(bam[[1]])
# Remove NA (unmapped) entries
bam_ranges <- bam_ranges[!is.na(bam_ranges$strand),]
# Convert BAM to Granges
bam_ranges <- GRanges(seqnames = bam_ranges$rname,
    ranges = IRanges(start =bam_ranges$pos , width = bam_ranges$qwidth),
    strand = bam_ranges$strand)

# Subset to chr7
bam_chr7 <-  bam_ranges[seqnames(bam_ranges) == "7"]
bam_chr7 <-  keepSeqlevels(bam_chr7, "7")

# Convert amplicons to Granges
amplicon_ranges <- GRanges(
     seqnames = Rle("7", 4),
     ranges = IRanges(start=c(55242379,55241584, 55259375 ,55248957 ),
     end = c(55242574, 55241796, 55259589, 55249194)),
     strand = Rle(strand("+"), 4)
     )

# Find Overlapping regions
(overlaps <- findOverlaps(bam_chr7, amplicon_ranges))
## Hits object with 496910 hits and 0 metadata columns:
##            queryHits subjectHits
##            <integer>   <integer>
##        [1]       242           2
##        [2]       243           2
##        [3]       244           2
##        [4]       284           2
##        [5]       285           2
##        ...       ...         ...
##   [496906]    992756           3
##   [496907]    992757           3
##   [496908]    992758           3
##   [496909]    992759           3
##   [496910]    992761           3
##   -------
##   queryLength: 993192 / subjectLength: 4
```

496 910 reads were aligned to coordinates overlapping with at least one of the four amplicons of interest, corresponding to 50% of all reads aligned to chromosome 7.

3. **What is the percentage of reads that were mapped to the human genome? Can you give some comments on the unmapped reads?**

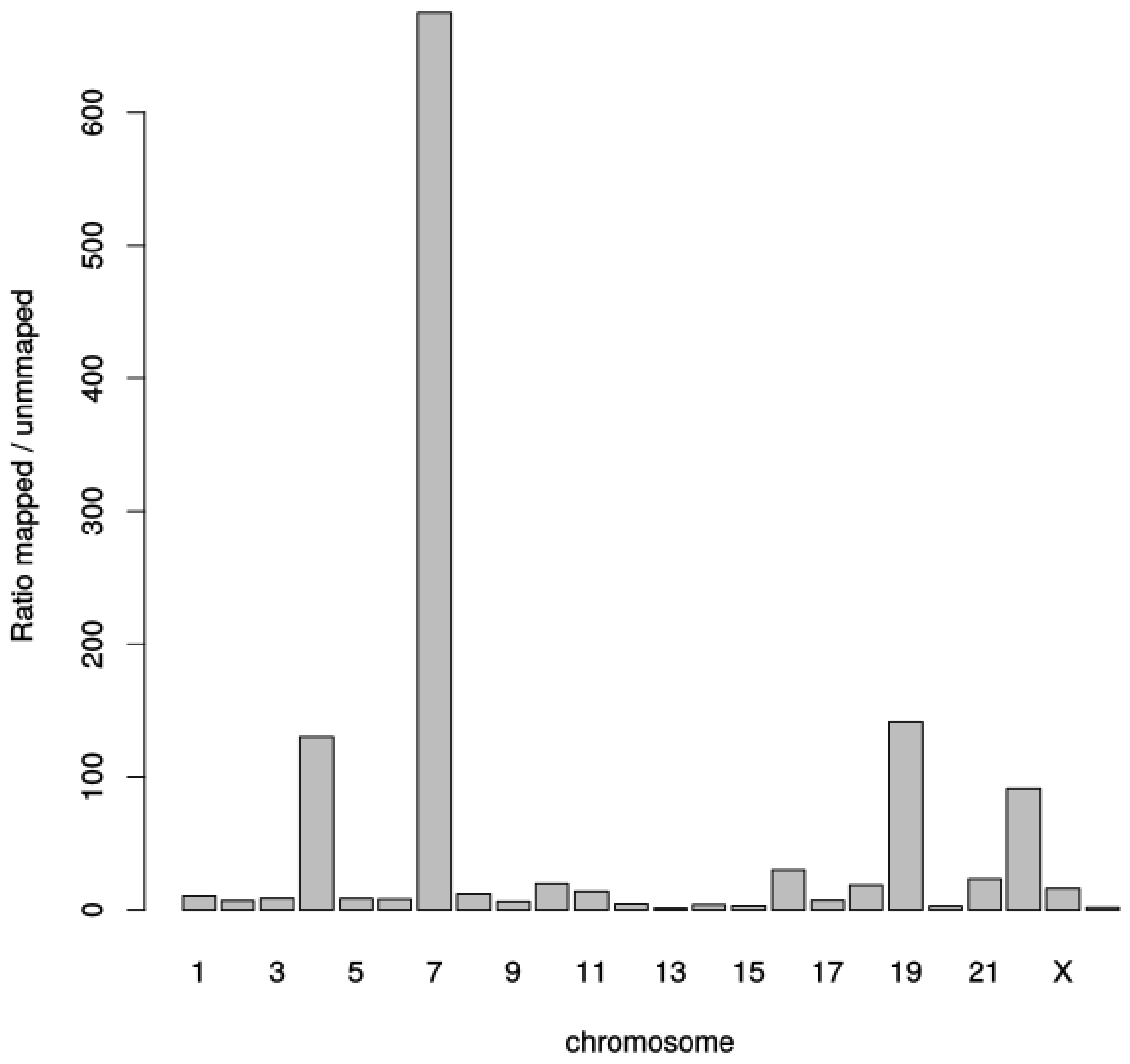
Answer: 1 052 966 of the 1 150 004 reads were aligned to one of the 24 human chromosomes, corresponding to 91% alignment to the genome.

Confirmation: Similarly to the previous question, I investigated the BAM alignment statistics to find how many of the reads were mapped to the human genome.

```
# Check available seqnames
bam_stats$seqnames
##  [1] 1          2          3          4          5          6
##  [7] 7          8          9          10         11         12
## [13] 13         14         15         16         17         18
## [19] 19         20         21         22         X          Y
## [25] MT         GL000191.1 GL000192.1 GL000193.1 GL000194.1 GL000195.1
## [31] GL000196.1 GL000197.1 GL000198.1 GL000199.1 GL000200.1 GL000201.1
## [37] GL000202.1 GL000203.1 GL000204.1 GL000205.1 GL000206.1 GL000207.1
## [43] GL000208.1 GL000209.1 GL000210.1 GL000211.1 GL000212.1 GL000213.1
## [49] GL000214.1 GL000215.1 GL000216.1 GL000217.1 GL000218.1 GL000219.1
## [55] GL000220.1 GL000221.1 GL000222.1 GL000223.1 GL000224.1 GL000225.1
## [61] GL000226.1 GL000227.1 GL000228.1 GL000229.1 GL000230.1 GL000231.1
## [67] GL000232.1 GL000233.1 GL000234.1 GL000235.1 GL000236.1 GL000237.1
## [73] GL000238.1 GL000239.1 GL000240.1 GL000241.1 GL000242.1 GL000243.1
## [79] GL000244.1 GL000245.1 GL000246.1 GL000247.1 GL000248.1 GL000249.1
## 84 Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 ... GL000249.1

# Autosomes
sum(bam_stats[bam_stats$seqnames %in% 1:22, "mapped"])
## [1] 1051084

# Sex chromosomes
sum(bam_stats[bam_stats$seqnames %in% c("X","Y"), "mapped"])
## [1] 1882
```

1 052 966 reads aligned to either the human sex chromosomes or the autosomes. In respect to the unmapped reads, I examined how the number of these relate to the number of mapped reads.

```
mapped_count <- bam_stats$mapped[1:24]
unmapped_count <- bam_stats$unmapped[1:24]

barplot(mapped_count/unmapped_count, names.arg = bam_stats$seqnames[1:24],
        xlab = "chromosome", ylab = "Ratio mapped / unmmaped")
```



,

From the barplot we can observe a low ratio between mapped and unmapped reads is present as *background* throughout the genome. However, the ratio on chromosome 7 is much larger than for the remaining genome, suggesting that these reads have been mapped to this chromosome successfully. Interestingly, a high ratio is also observed for other chromosomes, such as chromosome 4 and 19. I used the UCSC Genome Browser Blat tool, this time to find potential genomic sites with hight similarity to any of the expected amplicons. Complete identity was found between:

- Amplicon 1 and a sequence in chromosome 15
- Amplicon 2 and sequences in chromosomes 15 and 19
- Amplicon 3 and sequences in chromosomes 1, 3, 4 and 11
- Amplicon 4 and sequences in chromosomes 1, 9, 11, 12 and 14.

Consequently, the higher ratio of mapped reads in chromosome 4 and 19 may be the results of mis-alignment to similar sequences found elsewhere in the genome.

4. **What is the coverage of each amplicons? Could you find any explanations or clues why**

the coverage varies among amplicons, especially for those with big differences?

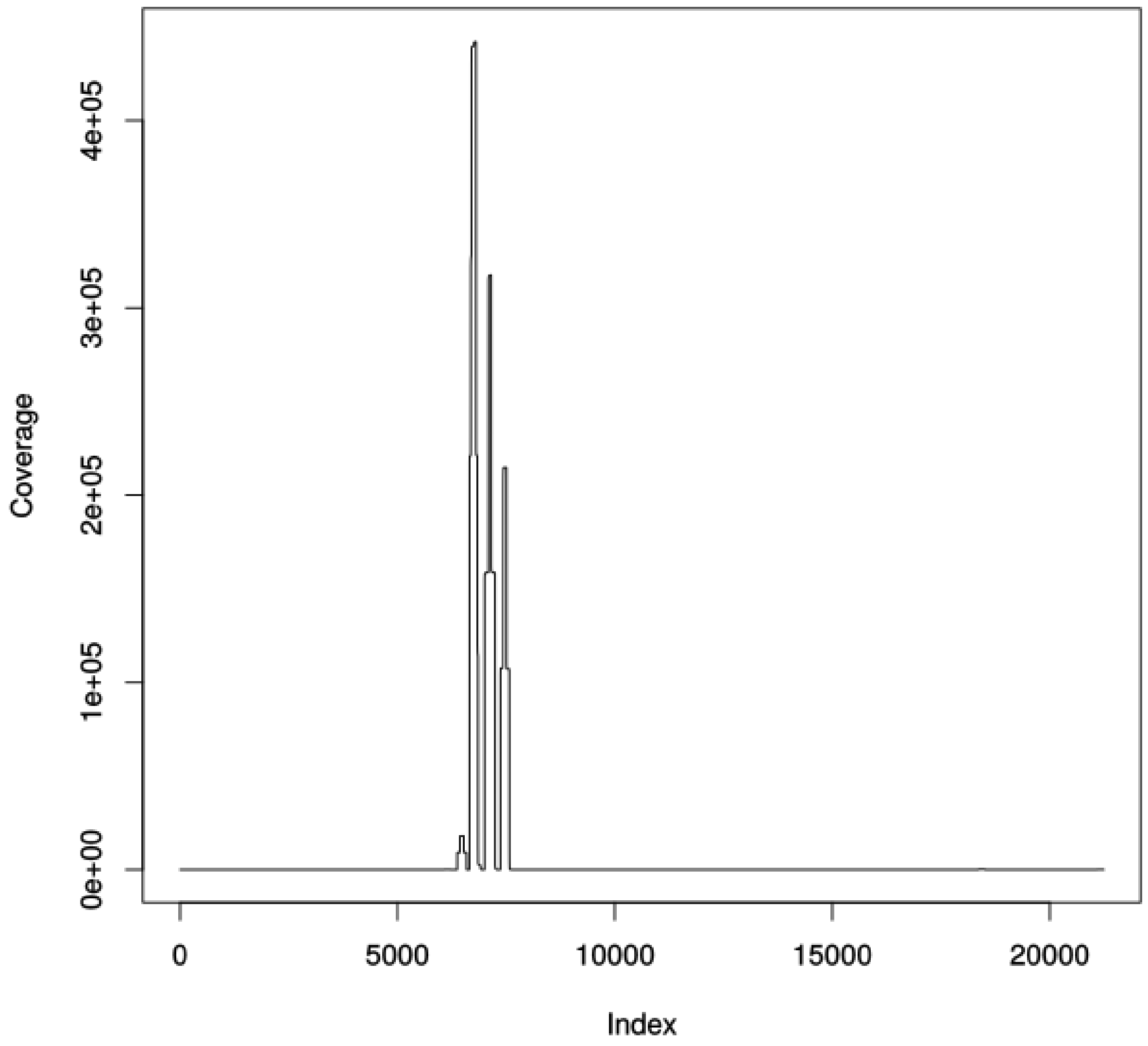
Answer: The coverage of sequencing varies for each amplicon. When normalized to the amplicon size

- Amplicon 1 shows 1 129 reads per base pair
- Amplicon 2 shows 42 reads per base pair
- Amplicon 3 shows 500 reads per base pair
- Amplicon 4 shows 667 reads per base pair.

Common reasons for variability in coverage include sequence structural properties (*e.g.* repeats, GC content) and PCR bias. I found no evidence of major intrinsic differences between sequences but there could be differences in PCR amplification efficiency during the target enrichment step, which might have been exponentially amplified. However, it could also be the case that a portion of amplicon 2 is deleted in the genome in study, thus explaining such a dramatic decrease in coverage.

Confirmation: Using the previous chromosome ranges obtained from the BAM file, I estimated the overall coverage across chromosome 7. I removed the regions with no mapped read to facilitate the analysis of the resulting plot.

```r
# Chr7 read coverage
cov <- coverage(bam_chr7)[[1]]
# Plot excluding regions of 0 coverage for simplification
plot(cov[cov !=0], type ="l", ylab = "Coverage")
```



,

```
# Check if the peak match expected
position <- 0
regions <- runLength(cov >10000)

for (idx in seq(1, length(regions)-1,2)){
  amplicon <- amplicon_ranges[
    ifelse(idx ==1 , 2,
           ifelse(idx == 3 , 2,
                  ifelse(idx== 5, 4, 3)))
    ]
  cat("Amplicon expected range: ", start(amplicon), "-", end(amplicon), "\n")
  position <- position + regions[idx]
  cat("Amplicon starting position: ", position, "\n")
  position <- position + regions[idx+1]
  cat("Amplicon ending position: ", position, "\n\n")
}
## Amplicon expected range:  55241584 - 55241796
## Amplicon starting position:  55241646
## Amplicon ending position:  55241734
##
## Amplicon expected range:  55241584 - 55241796
## Amplicon starting position:  55242378
## Amplicon ending position:  55242576
##
## Amplicon expected range:  55248957 - 55249194
## Amplicon starting position:  55248956
## Amplicon ending position:  55249195
##
## Amplicon expected range:  55259375 - 55259589
## Amplicon starting position:  55259374
## Amplicon ending position:  55259591

# Get numbers of reads per amplicon, normalized to amplicon size  (using the previous overlap)
for (amplicon in 1:4){
  amp_ranges <- bam_chr7[queryHits(overlaps)[subjectHits(overlaps) == amplicon]]
  amp_coverage <- length(amp_ranges) / width(amplicon_ranges)[amplicon]
  cat("Number of reads per size of ", amplicon, ":", amp_coverage , "\n")

}
## Number of reads per size of  1 : 1129.291
## Number of reads per size of  2 : 42.31925
## Number of reads per size of  3 : 500.4233
## Number of reads per size of  4 : 667.916
```

The four amplicon regions show the highest coverage for chromosome 7. Amplicon 1 (second peak) shows the highest coverage, followed by amplicon 4 and 3 (third and fourth peak respectively). Amplicon 2 (first peak), shows a very reduced coverage when compared to the remaining amplicons. Differences in coverage generally result from intrinsic sequence properties or PCR bias. In order to understand why amplicon 2 shows such a reduced coverage I looked at the GC ratio of each amplicon as this could influence the read coverage.

```
# GC content
letterFrequency(amplicons, "GC")/ width(amplicons)
##            G|C
## [1,] 0.4795918
## [2,] 0.5727700
## [3,] 0.5348837
## [4,] 0.5966387
```

I have additionally checked the amplicon sequences for repeats and low complexity sequences using Repeat-Masker

```
#RepeatMasker version open-4.0.6

#Checking for E. coli insertion elements
#identifying Simple Repeats in batch 1 of 1
#identifying full-length ALUs in batch 1 of 1
#identifying full-length interspersed repeats in batch 1 of 1
#identifying remaining ALUs in batch 1 of 1
```

```
#identifying most interspersed repeats in batch 1 of 1
#identifying long interspersed repeats in batch 1 of 1
#identifying ancient repeats in batch 1 of 1
#identifying retrovirus-like sequences in batch 1 of 1
#identifying Simple Repeats in batch 1 of 1

#No repetitive sequences were detected in /usr/local/rmserver/tmp/
```

There does not appear to be any clear differences in the amplicon 2 sequence that would explain the observed reduced coverage. I examined the PCR primers using the primer-Blast tool in order to understand if the primer pairs could be influencing the sequence target enrichment.

```
#Primer pair 1
#Forward primer - Tm: 66.07 GC%: 46.67
#Reverse primer - Tm: 71.12 GC%: 56.67

#Primer pair 2
#Forward primer - Tm: 70.87 GC%: 58.62
#Reverse primer - Tm: 77.84 GC%: 73.33

#Primer pair 3
#Forward primer - Tm: 69.56 GC%: 53.33
#Reverse primer - Tm: 58.09 GC%: 47.62

#Primer pair 4
#Forward primer - Tm: 72.30 GC%: 64.29
#Reverse primer - Tm: 58.49 GC%: 50.00
```

The reverse pair 2 appears to have a very high CG content (73%) and high melting temperature (TM) which may reduce its amplification efficiency. Additionally, reverse primers 3 and 4 have a TMs below 60C, when all the remaining primers have a Tm around 70C. These differences could result in lower efficiency during the PCR enrichment step. However, this can only be confirmed by estimating PCR efficiencies experimentally. Finally, because amplicon 2 coverage is so reduced, it could also be the case that it corresponds to a real result and amplicon 2 could be deleted from the genome being studied.

5. **What variants/mutations can you identify from this data? How can you evaluate which variants is more real than others?**

Answer: Variants/mutations were identified across the genome with 38% of all variants being called on chromosome 7. Of these, all were mapped to the amplification of interest. The only variant identified predicted to be non-synonymous was the 15 bp deletionlocated to the amplicon 1, which is expected to result in the loss of glu-leu-arg-glu-ala at the protein level of the EGFR transcripts containing the corresponding exon. 4 of the 15 bases associated with the identified deletion were matched to the dbSNP reference database and variants in these positions are likely pathogenic.

Besides testing functional effects and reference databases, to evaluate which variants are more likely to be real, one could additionally compare various calling methodologies (*e.g.* Freebayes, GATK), as each method is known to show bias to identify particular types of variants (*e.g.* SNPs, indels).

Confirmation: Many variant calling software are available. I have used samtools pileup here for simplicity. Variants were called if they had a minimum base and mapping quality of 20 and a minimum depth of 500, including deletions and insertions.

```
variants <- pileup("sequenceFile/aln_sorted.bam",
  index = "sequenceFile/aln.bam.bai",
  pileupParam=PileupParam(
    max_depth=50000,
    min_base_quality=20,
    min_mapq=20,
    min_nucleotide_depth=500,
    include_deletions=TRUE,
    include_insertions=TRUE,
    distinguish_strands=FALSE))

# Counts of identified variants per chromosome
table(variants$seqnames)[1:24]
##
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   0  33   0 191   0   0 700  40   0   0   0   0   0   0   0 160   0   0
##  19  20  21  22   X   Y
## 207   0   0 280 202   0
# Percentages of identified variants per chromosome
table(variants$seqnames)[1:24]/ sum(table(variants$seqnames)) * 100
##
##         1         2         3         4         5         6         7
##  0.000000  1.820188  0.000000 10.535025  0.000000  0.000000 38.610039
##         8         9        10        11        12        13        14
##  2.206288  0.000000  0.000000  0.000000  0.000000  0.000000  0.000000
##        15        16        17        18        19        20        21
##  0.000000  8.825152  0.000000  0.000000 11.417540  0.000000  0.000000
##        22         X         Y
## 15.444015 11.141754  0.000000

# Subset to variants overlapping mapping to chromosome 7
variants <- variants[variants$seq == "7",]

# Subset to variants overlapping the amplicons of interest
variants <- GRanges(
     seqnames = variants$seqnames,
     ranges = IRanges(start=variants$pos, width =1),
     alt = variants$nucleotide,
     freq = variants$count
     )
hits <- findOverlaps(variants, amplicon_ranges)
variants <- variants[queryHits(hits)]
variants$amplicon <- paste("Amplicon", subjectHits(hits))
variants
## GRanges object with 700 ranges and 3 metadata columns:
##         seqnames               ranges strand |      alt      freq
##            <Rle>            <IRanges>  <Rle> | <factor> <integer>
##     [1]        7 [55241584, 55241584]      * |        C      8845
##     [2]        7 [55241585, 55241585]      * |        C      8769
##     [3]        7 [55241586, 55241586]      * |        C      8875
##     [4]        7 [55241587, 55241587]      * |        T      8746
##     [5]        7 [55241588, 55241588]      * |        T      8915
##     ...      ...                  ...    ... .      ...       ...
##   [696]        7 [55259522, 55259522]      * |        A     46734
##   [697]        7 [55259523, 55259523]      * |        C     47431
##   [698]        7 [55259524, 55259524]      * |        T     47761
##   [699]        7 [55259525, 55259525]      * |        G     47882
##   [700]        7 [55259526, 55259526]      * |        C       520
##            amplicon
##         <character>
##     [1]  Amplicon 2
##     [2]  Amplicon 2
##     [3]  Amplicon 2
##     [4]  Amplicon 2
##     [5]  Amplicon 2
##     ...         ...
##   [696]  Amplicon 3
##   [697]  Amplicon 3
##   [698]  Amplicon 3
##   [699]  Amplicon 3
##   [700]  Amplicon 3
##   -------
##   seqinfo: 84 sequences from an unspecified genome; no seqlengths

# Check distribution of variants
table(variants$amplicon)
##
## Amplicon 1 Amplicon 2 Amplicon 3 Amplicon 4
##        181        213        152        154
# Check distribution of deletions
table(variants[variants$alt == "-"]$amplicon)
##
## Amplicon 1 Amplicon 4
##         15          1
```

700 variants were identified given the previous parameters, of which 16 corresponded to deletions. Amplicon 2 shows the most variants identified, which could be associated with the low coverage previously noticed, even though no deletions were found for this region. Since the amplicons of interest map to EGFR exons, it is interesting to understand if these variants can have functional effects at the protein level.

```
# Load required package
library(VariantAnnotation)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(BSgenome.Hsapiens.UCSC.hg19)
library(SNPlocs.Hsapiens.dbSNP144.GRCh37)
```

```
# Match seqnames
variants <- keepSeqlevels(variants, "7")
seqlevels(variants, force=TRUE) <- c("7"="chr7")

# Predict coding effects
coding <- predictCoding(variants, TxDb.Hsapiens.UCSC.hg19.knownGene, Hsapiens,
  DNAStringSet(droplevels(variants$alt)))
## Warning in .predictCodingGRangesList(query, cache[["cdsbytx"]],
## seqSource, : 'varAllele' values with 'N', '.', '+' or '-' were not
## translated

# Distribution of coding effects predicted
table(coding$CONSEQUENCE)
##
## not translated     synonymous
##             75           2532
```

```
# Check 'not translated' variants
unique(as.data.frame(coding)[coding$CONSEQUENCE=="not translated",c("start", "amplicon", "REFCODON", "VA
##          start   amplicon REFCODON VARCODON
## 871  55242466 Amplicon 1      GAA      -AA
## 881  55242467 Amplicon 1      GAA      G-A
## 891  55242468 Amplicon 1      GAA      GA-
## 901  55242469 Amplicon 1      TTA      -TA
## 911  55242470 Amplicon 1      TTA      T-A
## 921  55242471 Amplicon 1      TTA      TT-
## 931  55242472 Amplicon 1      AGA      -GA
## 941  55242473 Amplicon 1      AGA      A-A
## 951  55242474 Amplicon 1      AGA      AG-
## 961  55242475 Amplicon 1      GAA      -AA
## 971  55242476 Amplicon 1      GAA      G-A
## 981  55242477 Amplicon 1      GAA      GA-
## 991  55242478 Amplicon 1      GCA      -CA
## 1001 55242479 Amplicon 1      GCA      G-A
## 1011 55242480 Amplicon 1      GCA      GC-
```

Most of the called variants (97%) are expected to be synonymous mutations and thus have no effect on the gene function. However, 75 transcripts show potential ‘not translated’ phenotype. These 75 transcripts are the result of 15 deleted bases found on amplicon 1. Looking carefully at these bases, it would appears that the region from 55 242 466 to 55 242 480 corresponds to an unique deletion of GAATTAAGAGAAGCA on amplicon 1, leading to the loss of glu-leu-arg-glu-ala at the protein level.

Comparisons of the identified variants with curated databases such as dbSNP provides additional confidence to the results, as variants known to occur in the population are more likely to be true positives.

```
seqlevels(variants, force=TRUE) <- c("chr7"="ch7")
(snps <- snpsByOverlaps(SNPlocs.Hsapiens.dbSNP144.GRCh37, variants))
## GPos object with 172 positions and 2 metadata columns:
##         seqnames       pos strand |   RefSNP_id alleles_as_ambig
##            <Rle> <integer>  <Rle> | <character>      <character>
##     [1]      ch7  55241584      + | rs751906002                S
##     [2]      ch7  55241589      + | rs751132626                K
##     [3]      ch7  55241590      + | rs754843361                W
##     [4]      ch7  55241603      + | rs781261607                S
##     [5]      ch7  55241607      + | rs184220351                S
##     ...      ...       ...    ... .         ...              ...
##   [168]      ch7  55259515      + | rs121434568                K
##   [169]      ch7  55259516      + | rs397517129                D
##   [170]      ch7  55259518      + | rs763666690                Y
##   [171]      ch7  55259522      + | rs397517130                W
##   [172]      ch7  55259524      + | rs121913444                D
##   -------
##   seqinfo: 25 sequences (1 circular) from GRCh37.p13 genome

# Check distribution of matched variants
table(variants[start(variants) %in% pos(snps)]$amplicon)
##
## Amplicon 1 Amplicon 2 Amplicon 3 Amplicon 4
##         52         42         33         49

# Check types of matched variants
```

```
table(variants[start(variants) %in% pos(snps)]$alt)
##
##  A  C  G  T  N  =  -  +
## 21 64 65 22  0  0  4  0

# Check deletions
(snp_dels <- snps[pos(snps) %in% start(variants[variants$alt == "-"])])
## GPos object with 4 positions and 2 metadata columns:
##       seqnames       pos strand |   RefSNP_id alleles_as_ambig
##          <Rle> <integer>  <Rle> | <character>      <character>
##   [1]      ch7  55242466      + | rs121913427                V
##   [2]      ch7  55242469      + | rs397517095                Y
##   [3]      ch7  55242470      + | rs397517097                Y
##   [4]      ch7  55242478      + | rs121913229                S
##   -------
##   seqinfo: 25 sequences (1 circular) from GRCh37.p13 genome

# Print links to dbSNP
paste0("https://www.ncbi.nlm.nih.gov/projects/SNP/snp_ref.cgi?rs=",
  snp_dels$RefSNP_id)
## [1] "https://www.ncbi.nlm.nih.gov/projects/SNP/snp_ref.cgi?rs=rs121913427"
## [2] "https://www.ncbi.nlm.nih.gov/projects/SNP/snp_ref.cgi?rs=rs397517095"
## [3] "https://www.ncbi.nlm.nih.gov/projects/SNP/snp_ref.cgi?rs=rs397517097"
## [4] "https://www.ncbi.nlm.nih.gov/projects/SNP/snp_ref.cgi?rs=rs121913229"
```

172 of the 700 variants were matched to the SNP (single nucleotide polymorphism) database, including 4 of the 15 bases identified as deleted. 3 of these reference SNPs have a Clinical Significance of *Likely pathogenic, untested allele* and one *Uncertain significance allele.*

Notes

```
sessionInfo()
```

```
## R version 3.3.3 (2017-03-06)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.1 LTS
##
## locale:
##  [1] LC_CTYPE=en_GB.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_GB.UTF-8        LC_COLLATE=en_GB.UTF-8
##  [5] LC_MONETARY=en_GB.UTF-8    LC_MESSAGES=en_GB.UTF-8
##  [7] LC_PAPER=en_GB.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_GB.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] SNPlocs.Hsapiens.dbSNP144.GRCh37_0.99.11
##  [2] BSgenome.Hsapiens.UCSC.hg19_1.4.0
##  [3] BSgenome_1.42.0
##  [4] rtracklayer_1.34.2
##  [5] TxDb.Hsapiens.UCSC.hg19.knownGene_3.2.2
##  [6] GenomicFeatures_1.26.3
##  [7] AnnotationDbi_1.36.2
##  [8] VariantAnnotation_1.20.2
##  [9] dplyr_0.5.0
## [10] ShortRead_1.32.0
## [11] GenomicAlignments_1.10.0
## [12] SummarizedExperiment_1.4.0
## [13] Biobase_2.34.0
## [14] Rsamtools_1.26.1
## [15] GenomicRanges_1.26.3
## [16] GenomeInfoDb_1.10.3
## [17] Biostrings_2.42.1
## [18] XVector_0.14.0
## [19] IRanges_2.8.1
## [20] S4Vectors_0.12.1
## [21] BiocParallel_1.8.1
## [22] BiocGenerics_0.20.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.10        highr_0.6           RColorBrewer_1.1-2
##  [4] bitops_1.0-6        tools_3.3.3         zlibbioc_1.20.0
##  [7] biomaRt_2.30.0      digest_0.6.12       evaluate_0.10
## [10] RSQLite_1.1-2       memoise_1.0.0       tibble_1.2
## [13] lattice_0.20-34     Matrix_1.2-8        DBI_0.5-1
## [16] stringr_1.2.0       hwriter_1.3.2       knitr_1.15.1
## [19] grid_3.3.3          R6_2.2.1            XML_3.98-1.5
## [22] latticeExtra_0.6-28 magrittr_1.5        codetools_0.2-15
## [25] assertthat_0.1      stringi_1.1.2       lazyeval_0.2.0
## [28] RCurl_1.95-4.8
```

To build pdf run:

knitr::knit("sophia_script.Rmd")
# Fine tune the .md file as required
rmarkdown::render("sophia_script.md", output_format = "pdf_document", output_file = "SophiaGenetics_CNreport.pdf")